AFRL-RV-PS-
TR-2012-0248

AFRL-RV-PS-
TR-2012-0248

HIERARCHICAL BIO-INSPIRED COOPERATIVE CONTROL FOR NONLINEAR DYNAMICAL SYSTEMS AND HARDWARE DEMONSTRATION

Yunjun Xu

University of Central Florida
4000 Central Florida Blvd., Engineering Building 1
Mechanical, Materials, and Aerospace Engineering
Orlando, FL 32765

3 Apr 2013

Final Report

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION IS UNLIMITED.

AIR FORCE RESEARCH LABORATORY
Space Vehicles Directorate
3550 Aberdeen Ave SE
AIR FORCE MATERIEL COMMAND
KIRTLAND AIR FORCE BASE, NM 87117-5776

DTIC COPY
NOTICE AND SIGNATURE PAGE

AFRL-RV-PS-TR-2012-0248 HAS BEEN REVIEWED AND IS APPROVED FOR PUBLICATION IN ACCORDANCE WITH ASSIGNED DISTRIBUTION STATEMENT

//SIGNED//
KHANH PHAM
Program Manager

//SIGNED//
PAUL HAUSGEN
Technical Advisor, Spacecraft Component Technology Branch

//SIGNED//
CHRISTOPHER D. STOIK, Lt Col, USAF
Deputy Chief, Spacecraft Technology Division
Space Vehicles Directorate

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YY)*
03-04-2013

2. REPORT TYPE
Final Report

3. DATES COVERED *(From - To)*
19 Aug 2011 – 17 Feb 2013

4. TITLE AND SUBTITLE
Hierarchical Bio-Inspired Cooperative Control for Nonlinear Dynamical Systems and Hardware Demonstration

5a. CONTRACT NUMBER
FA9453-11-1-0305

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER
62601F

6. AUTHOR(S)
Yunjun Xu

5d. PROJECT NUMBER
8809

5e. TASK NUMBER
PPM00013329

5f. WORK UNIT NUMBER
EF005648

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
University of Central Florida
4000 Central Florida Blvd., Engineering Building 1
Mechanical, Materials, and Aerospace Engineering
Orlando, FL 32765

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
Air Force Research Laboratory
Space Vehicles Directorate
3550 Aberdeen Ave., SE
Kirtland AFB, NM 87117-5776

10. SPONSOR/MONITOR'S ACRONYM(S)
AFRL/RVSV

11. SPONSOR/MONITOR'S REPORT NUMBER(S)
AFRL-RV-PS-TR-2012-0248

12. DISTRIBUTION / AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES

14. ABSTRACT
This report lists all the research results that have been achieved during the funding period for robot trajectory planning problems and testbed: (i) research goals and milestones, (ii) hardware/software design, integration, and validation, (iii) bio-inspired trajectory planning algorithm, (iv) robot localization and obstacle detection, (v) simulation validation and hardware demonstration. All the research objectives have been accomplished.

15. SUBJECT TERMS
Bio-inspired trajectory planning, virtual motion camouflage, cooperative control, LEGO robotic testbed, non-linear dynamics

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
					Khanh Pham
a. REPORT	b. ABSTRACT	c. THIS PAGE			19b. TELEPHONE NUMBER *(include area code)*
Unclassified	Unclassified	Unclassified	Unlimited	30	

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std. 239.18

(This page intentionally left blank)

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS AND DISCLAIMER

Acknowledgement

We would like thank the support from the Air Force Research Laboratory, Space Vehicle Directorate through Grant #FA94531110305 (Program Manager: Dr. Khanh Pham). Also we would like to thank the following students from the Dynamics and Controls Laboratory at the University of Central Florida for their technical contributions: Gareth Basset, Robert SiVilli, and Charles Remeikas. We also appreciate our subcontract Dr. Ming Xin (the Mississippi State University) for helping us with the top level algorithm.

Disclaimer

(This page intentionally left blank)

1 SUMMARY

Rapid trajectory planning and control for single or cooperative vehicles is an area of special interest in the field of autonomous robotics, and new research and development of innovative algorithms for these systems are at an all-time high due to advances in computing power, communication, and sensors.

In this project, a "divide-and-conquer" based hierarchical approach empowered by the virtual motion camouflage based trajectory planning algorithm is investigated to tackle the challenges experienced in the cooperative planning problems of dynamical systems. The salient features of the proposed algorithm are the low computational cost and scalability. Through this method, the problem dimension of the achieved nonlinear programming problem is significantly reduced, such that it can be solved much quicker, while still being able to find the optimal solution. The method is supplemented by two perturbation techniques and augmented by the wavefront algorithm, which are used to generate good initial guesses for the optimization process and significantly enhance the success rate. In addition, the bio-inspired approach proposed in the bottom level can easily address nonlinear dynamics, conflict resolution, obstacle avoidance, etc. For the vision processing aspect, a template-based predictive search algorithm is applied to process the images obtained through a low-cost webcam vision system, which is used to monitor the testbed environment. Also a user-friendly graphical interface is developed such that the functionalities of the webcam, robots, and optimizations are automated. The capabilities of this new algorithm have been successfully demonstrated in the low cost robot platforms constructed at the University of Central Florida and the Air Force Research Laboratory – Space Vehicle Directorate.

In the final report, the following items will be discussed in details: hierarchical control framework, bio-inspired optimal trajectory planning method, robot vehicle dynamics and obstacle avoidance, hardware platform, robot localization and obstacle detection, low level robot path tracking, automatic robot parameter calibration, and software architecture, simulation results, and hardware demonstrations.

2 INTRODUCTION

Cooperative control is crucial for networked dynamical systems (e.g. distributed underwater, ground, aerial, and/or space vehicles) to work effectively in dynamically changing and cluttered environments [1].

To date, the majority of existing algorithms are developed based on simple models such as single or double integrators that a fast response can be achieved in a feedback form and that many useful properties within a framework of graph theories, linear system theories, and matrix algebra can be easily applied [2][3][4]. However, real dynamic systems, such as spacecraft and satellites, have much more complex nonlinear dynamics, or are heterogeneous with possibly each vehicle having different dynamical characteristics, nonlinearities, and constraints, and may operate in adverse and uncertain environments.

On the other hand, to date, developing a cooperative optimal control strategy that considers all the above mentioned constraints is still a very challenging task [5] from both theoretical and implementation perspectives. Most of the current approaches are computationally expensive and can only be used offline [6][7].

In this research, a hierarchical approach empowered by the virtual motion camouflage based trajectory planning method is introduced to tackle these challenges based on the divide-and-conquer strategy. The capabilities of this innovative approach are demonstrated in a low cost robot formation.

The proposed research has the following objectives:

O1: Hardware Demonstration: The new algorithm will be demonstrated in a group of low cost robots.

O2: Top-Level Optimal Distributed Control: an optimal formation algorithm will be investigated to guarantee the macro cooperative behaviors of the networked dynamic system under simplified double-integrator dynamics and considering obstacle avoidance.

O3: Bottom-Level Bio-Inspired Fast Planning Algorithm: A virtual motion camouflage based real-time algorithm will be investigated for each individual vehicle to optimize the trajectory neighboring the one produced in the top-level. The algorithm will address various types of real-world constraints that have been intentionally omitted in the top level.

O4: A low-cost robot testbed will be developed to validate the investigated new algorithms.

3 METHODS, ASSUMPTIONS, AND PROCEDURES

The research conducted is summarized as: hierarchical control framework, bio-inspired optimal trajectory planning method, robot vehicle dynamics and obstacle avoidance, hardware platform, robot localization and obstacle detection, low level robot path tracking, automatic robot parameter calibration, and software architecture, simulation results, and hardware demonstrations.

3.1 Hierarchical Framework

The hierarchical structure is illustrated in Fig. 1. Following the concept of "divide-and-conquer", two methods are combined to address cooperative planning challenges at the top level and the real-world constraints at the bottom level separately to tackle computational cost issues.

In this architecture (Fig. 1), the top level algorithm generates reference cooperative trajectories to be used by each vehicle at the bottom level over a time interval of $[t_k, t_{k+1}]$. The only information transmitted from the top level to the bottom level of each vehicle is the reference trajectory (position). The bottom level algorithm optimizes the actual trajectory considering all realistic constraints and dynamics. Control commands generated by the bottom level are the ones actually used for vehicle level tracking control in this hierarchical approach.

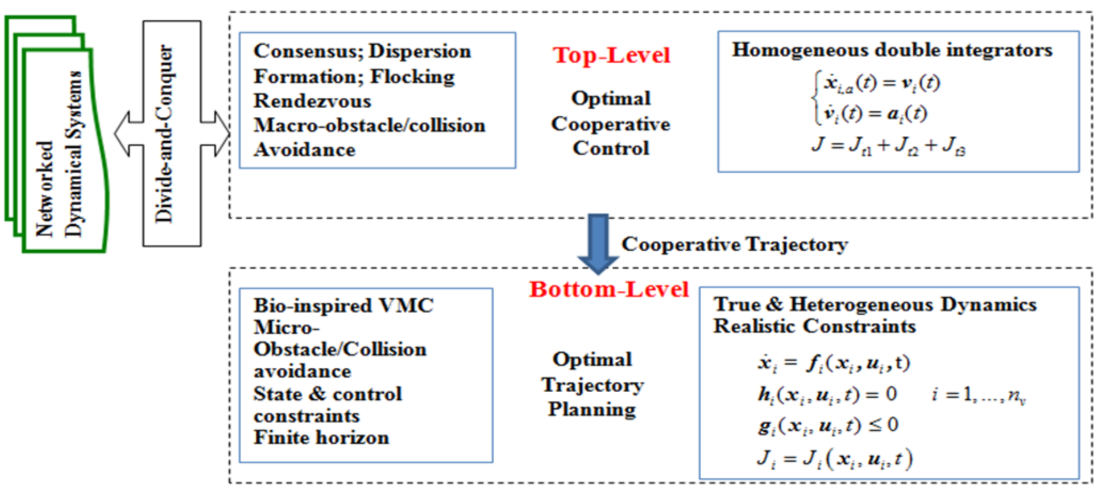

Figure 1. Hierarchical Structure of the cooperative control [8].

3.2 Bio-Inspired Optimal Trajectory Planning / Re-planning Algorithm

The virtual motion camouflage (VMC) inspired varying subspace optimal trajectory planning method has been customized in this research to be two approaches: (1) the first approach (optimal) is used in the individual vehicle's trajectory planning [9]; and (2) the second approach (suboptimal) is used in the hierarchical cooperative approach. The main difference between these two approaches is: in (1) the virtual prey motion is represented via B-Spline curves and the parameters controlling the shape of the B-Spline curves are optimized; and in (2) the virtual prey motion is passed from the top level algorithm and itself will not be optimized.

The motivation of the VMC method and the analysis of the solution optimality can be found in [8][10]. For brevity, only the steps involved in this algorithm is listed here.

The motion camouflage (MC) strategy [11] is used by the male hoverfly (i.e. aggressor) to conceal its motion as seen by the female hoverfly (i.e. prey) in mating activities (Fig. 2). There are three variables determining the path of the aggressor: reference point $\boldsymbol{x}_r$, a one-dimensional time varying path control parameter (PCP) $v(t)$ [12], and the prey motion $\boldsymbol{x}_p(t)$. The aggressor can pick any reference point and PCP variables as long as its position satisfies the MC rule as

$$\boldsymbol{x}_a(t) = \boldsymbol{x}_r + v(t)[\boldsymbol{x}_p(t) - \boldsymbol{x}_r] \tag{1}$$

An interesting observation about this phenomenon is: the aggressor only moves along the path $\boldsymbol{x}_a(t)$ in a subspace constructed by the prey motion $\boldsymbol{x}_p(t)$ and the reference point $\boldsymbol{x}_r$ according to Eq. (1) at any instance.

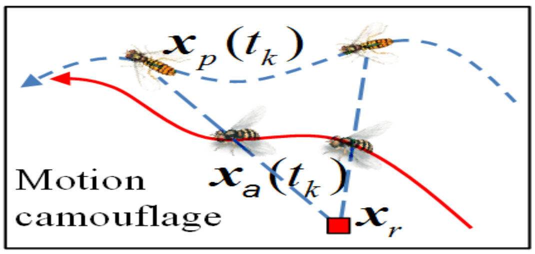

Figure 2. Motion camouflage

A typical trajectory planning and re-planning problem is to solve for the state $\boldsymbol{x}$ and control $\boldsymbol{u}$ to minimize a performance index $J(\boldsymbol{x},\boldsymbol{u})$, subject to the nonlinear dynamics $\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x},\boldsymbol{u})$, inequality constraints $\boldsymbol{g}(\boldsymbol{x},\boldsymbol{u}) \le 0$, and equality constraints $\boldsymbol{h}(\boldsymbol{x},\boldsymbol{u}) = 0$.

For many dynamical systems (like the ones tested in this research), the state vector can be separated as $\boldsymbol{x} = [\boldsymbol{x}_a^T, \boldsymbol{x}_{sr}^T]^T$, of which $\boldsymbol{x}_a \in \Re^{n_a \times 1}$ is the "position" state and the remaining states $\boldsymbol{x}_{sr}$ are called the "state rate". The VMC approach begins by defining the "position" part of the states as the aggressor motion. Via the MC strategy, the aggressor motion is limited in the subspace constructed according to Eq. (1). The reference point needs to be optimized. The derivatives of the "position" state $\boldsymbol{x}_a$ can be calculated via

$$\dot{\boldsymbol{x}}_a = \dot{v}\left(\boldsymbol{x}_p - \boldsymbol{x}_r\right) + v\dot{\boldsymbol{x}}_p \tag{2}$$

$$\ddot{\boldsymbol{x}}_a = \ddot{v}\left(\boldsymbol{x}_p - \boldsymbol{x}_r\right) + v\ddot{\boldsymbol{x}}_p + 2\dot{v}\dot{\boldsymbol{x}}_p \tag{3}$$

and so on. Through the dynamic inversion procedure, the "state rate" variables $\boldsymbol{x}_{sr}$ and the control input $\boldsymbol{u}$ can be represented as functions of the PCPs, virtual prey motion, reference point, and their corresponding derivatives.

After that, the PCP $v(t)$ is discretized using a high order discretization scheme, such as the Legendre-Gauss-Lobatto method [6]. The parameters to be optimized are the PCP nodes v_k, $k = 0,\ldots,N$, and the reference point $\mathbf{x}_r$. Further development in has shown that boundary conditions can be used to further reduce the number of the parameters to be optimized [9].

Method 1 (Used in Single Vehicle Control): The optimality of the solution obtained in the VMC formulation is determined by the varying subspace constructed by the virtual prey and the reference point. Therefore to enhance the solution optimality, the virtual prey motion and thus the subspace should also be varied and optimized. Here the virtual prey motion is approximated by B-Splines [13] as

$$x_{p,i}(t_k) = \sum_{j=0}^{n_{cp}} B_{j,d}(t_k) P_{i,j} \quad i = 1,\ldots,n_a \tag{4}$$

In Eq. (4), $n_{cp}+1$ is the number of control points used in the B-Spline representation. It is worth noting that as n_{cp} approaches infinite, the parameterized B-Spline curve will converge to the actual virtual prey motion. However, it is not practical to have a large n_{cp}. Therefore there is always a tradeoff between the computational cost and solution optimality. The n_{cp} is initially selected by users, and the software is programmed that if an initial guess of n_{cp} does not achieve a converged solution, this value will be perturbed a little bit. The derivatives of the virtual prey motion can be calculated by

$$\dot{x}_{p,i}(t_k) = \sum_{j=0}^{n_{cp}} \dot{B}_{j,d}(t_k) P_{i,j} \quad i = 1,\ldots,n_a \tag{5}$$

$$\ddot{x}_{p,i}(t_k) = \sum_{j=0}^{n_{cp}} \ddot{B}_{j,d}(t_k) P_{i,j} \quad i = 1,\ldots,n_a \tag{6}$$

and so on, where $P_{i,j}$, $i = 1,\ldots,n_a, j = 0,\ldots,n_{cp}$ are the control points that represent the shape of the virtual prey motion. $B_{j,d}(t_k)$, $j = 0,\ldots,n_{cp}$, are the d^{th} degree basis functions, and $\dot{B}_{j,d}(t_k)$ and $\ddot{B}_{j,d}$ are the first and second derivatives of the basis functions, respectively. The detailed equations in calculating those basis functions can be found in [13]. As described in [17], a wavefront algorithm [16] and a smoothing algorithm will be used to provide a collision free and smoothing prey motion.

Method 2 (Used in the Hierarchical Framework): The trajectory calculated by the top level planner will be regarded as the prey motion (not arbitrary selected), and the reference point will be regarded as optimization parameters. In this approach, the cooperative planning trajectory generation algorithm can be found in [8]. In the starting of each horizon, the top-level algorithm will propagate for this horizon. The cooperative path will be used as the prey motion, over which the bottom level will optimize the vehicles' trajectories.

3.3 Robot Vehicle Dynamics and Obstacle Avoidance

A simple nonlinear model is used in our testbed to represent two-wheel robot [14] as

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos\theta \\ \sin\theta \\ 0 \end{bmatrix} \upsilon + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} w \tag{7}$$

where the two wheels' midpoint $[x, y]^T$ is regarded as the "position" state and θ is the heading angle. Two control variables (related to the translational and rotational power) are involved as the speed v and the angular speed w, and they are respectively constrained by $|v| \le v_{\max}$ and $|w| \le w_{\max}$. **Obstacle avoidance**: obstacle avoidance is considered to be inequality constraints in the achieved nonlinear programming problem. For simplicity, all the obstacles are assumed to be circular and the inequality constraints are modeled as $\left(x - x_{i,obs}\right)^2 + \left(y - y_{i,obs}\right)^2 \ge r_{i,obs}^2$, in which $r_{i,obs}, x_{i,obs}$, and $y_{i,obs}$ are the radius and coordinate of the i^{th} obstacles respectively. Here all these values can be either detected by the vision processing algorithm in the hardware or predefined in the testbed.

3.4 Cooperative Planning in the Top Level

At the top level, the robot i is abstracted as a double-integrator:

$$\begin{cases} \dot{\boldsymbol{x}}_{i,a}(t) = \boldsymbol{v}_i(t) \\ \dot{\boldsymbol{v}}_i(t) = \boldsymbol{a}_i(t) \end{cases}, \quad i = 1, \ldots, n_v \tag{8}$$

in which vehicle i is assumed to be a point mass, and $\boldsymbol{x}_{i,a}$, $\boldsymbol{v}_i$, and $\boldsymbol{a}_i$ are the position, velocity, and acceleration vectors. The top level planner is to achieve a desired cooperative behavior, while avoiding obstacles characterized. Denoting the aggregate state and control variables at the top level as $\boldsymbol{X} = [\boldsymbol{x}_{1,a}^T, \cdots, \boldsymbol{x}_{n_v,a}^T, \boldsymbol{v}_1^T, \cdots, \boldsymbol{v}_{n_v}^T]^T$ and $\boldsymbol{U} = [\boldsymbol{a}_1^T, \cdots, \boldsymbol{a}_{n_v}^T]^T$. The cost function to be optimized at the top level is

$$J = J_{t1} + J_{t2} + J_{t3} \triangleq \int_0^\infty \Upsilon(\boldsymbol{X}, \boldsymbol{U}) dt \tag{9}$$

where J_{t1}, J_{t2}, and J_{t3} represent the costs associated with the formation, obstacle avoidance, and control effort, respectively. The top level cooperative planning solution can be found in [8].

3.5 Hardware Platform

The proposed algorithm is demonstrated in a robot formation control testbed as shown in Fig. 3. In this test scenario, several obstacles will be placed in the test area to validate the collision avoidance capability of the algorithm. The robots obtain their commands from the laptop and the laptop obtained the position and obstacle information from the overhead camera. The position of the robot can be obtained through the vision system monitoring the whole test area. The algorithms described will compute the optimal path in the laptop computer and then these commands will be transmitted via the Bluetooth (® Bluetooth word mark and logos are registered trademarks owned by Bluetooth SIG, Inc.) to the corresponding robots for them to follow.

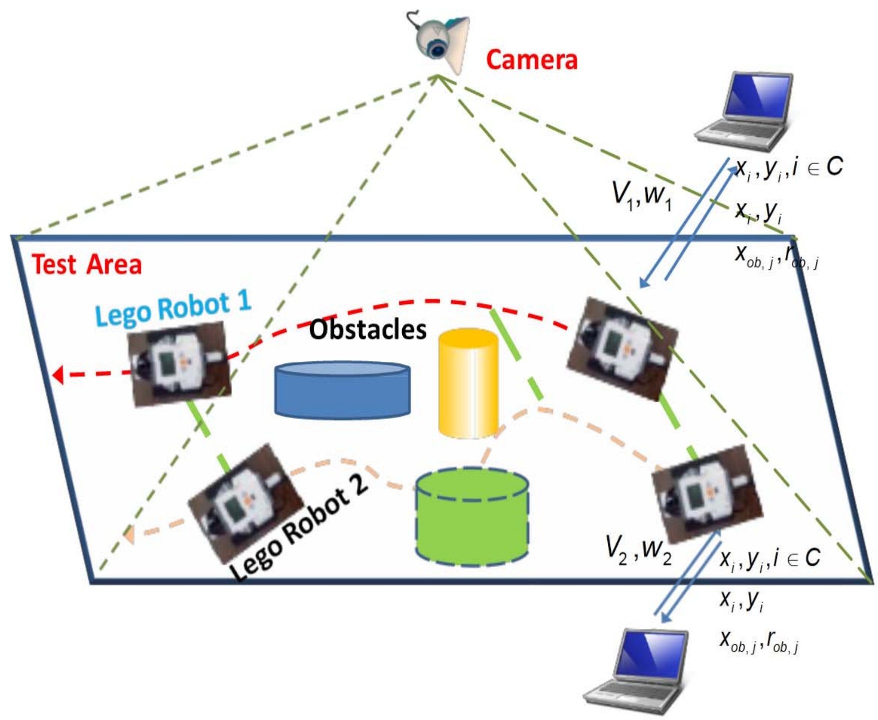

Figure 3. Robots formation control testbed

The architecture of the LEGO robots (® LEGO is a trademark and/or copyright of the LEGO Group) used in tests were based off the quick-start, tracked robot design centered around a NXT 'Brick.' This brick contains a 32-bit ARM7 microprocessor with 256 Kbytes Flash/64 Kbytes RAM, a 8-bit microprocessor with 4 Kbytes Flash/512 Byte RAM, four sensor input ports and 3 motor output ports. Each brick is also capable of Bluetooth wireless communication which was used to control each robot and obtain sensor information. Each robot's left and right tracks are separately controlled via singles from the NXT brick which is also where data is collected from the sensors before being sent to a central hub via Bluetooth (**Fig. 4** and **Fig. 5**).

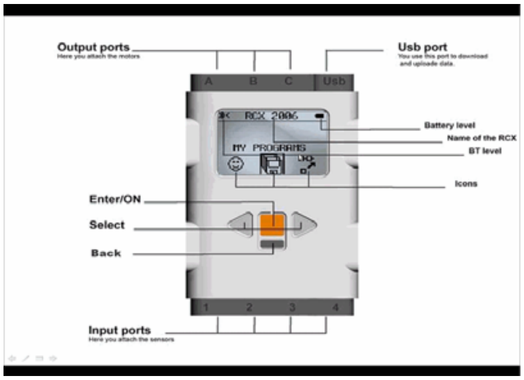

Figure 4. NXT brick is the brain of the robots and acts as the interface between each robot's sensors/motors and the upper level control.

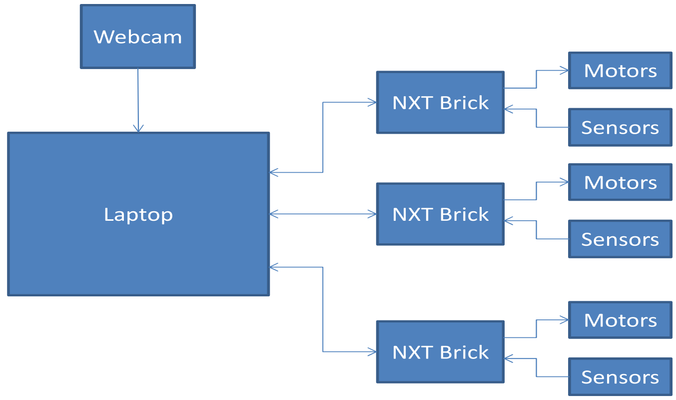

Figure 5. The communication diagram for three robots.

On-board sensors include a single axis gyroscope, three axis accelerometer, and compass. The accelerometer is capable of measuring accelerations up to +/-19.62 m/s with a resolution of 0.04905 m/s on each axis at 100 Hz. The gyroscope samples at 300 Hz with a resolution of 1 deg/s and a maximum +/- 360 degrees/s while the compass samples at 100 Hz with a resolution of 1 degree and ranges from 0 to 359 degrees. However in the interest of reducing delays in data collection, only the compass was used and an even that was in an auxiliary capacity to be discussed along with the vision system next.

Robot and obstacle information was determined through the webcam based vision system suspended over the test bed. Each robot sensor mounting was covered with two dots, one black and the other uniquely colored. The uniquely colored dot was used to differentiate each robot and the combination of the two was used for heading calculations.

The motor onboard of the robot will be regarded as the actuator. Varying the power input to the motor from 0% to 100% will give different speed of the robot. The building sensor of the motor can be accurate to +/1 degree.

The testbed configuration described above has the following advantages in the perspective of testing the proposed algorithm: (1) this platform is low cost; (2) the platform is compatible with other hardware platform and can communicate with them through USB or Bluetooth; and (3) the robot can talk to the algorithms written in MATLAB on the laptop through Bluetooth, which can significantly reduce the time in the software development cycle.

3.6 Robot Localization and Obstacle Detection

Robot localization and obstacle detection are accomplished through almost identical means using an overhead camera, capturing 480x640 resolution images at rates up to 30fps. In either case, a normalized cross correlation matrix (NCC) [18] $\gamma \in R^{480\times 640}$ is created using a grayscale image from the vision system, $H_{gs} \in R^{480\times640}$, and a grayscale template of the object of interest, $f \in R^{M\times N}$, such that

$$\gamma(u,v) = \frac{\sum_{x,y}\left[H_{gs}\left(x,y\right)-\bar{H}_{u,v}\right]\left[f\left(x-u,y-v\right)-\bar{f}\right]}{\left\{\sum_{x,y}\left[H_{gs}\left(x,y\right)-\bar{H}_{u,v}\right]^2\sum_{x,y}\left[f\left(x-y,y-v\right)-\bar{f}\right]^2\right\}^{0.5}} \tag{10}$$

where $\bar{f}$ is the mean of the grayscale template and $\overline{H}_{u,v}$ is the mean of the grayscale image under the template shifted to $[u,v]$.

Object locations are determined by the local maximums of γ passing a set tolerance. The results are illustrated in **Figure 6** and **Figure 7** for the obstacle detection and the robot localization, respectively. It is worth noting that because all obstacles currently used in the testbed are homogenous, this is all that is required to obtain the location and radius of the obstacles $[x_{obst}, y_{obst}, r_{obst}]_i, i = 1,...,n_{obst}$, where n_{obst} is the number of obstacles.

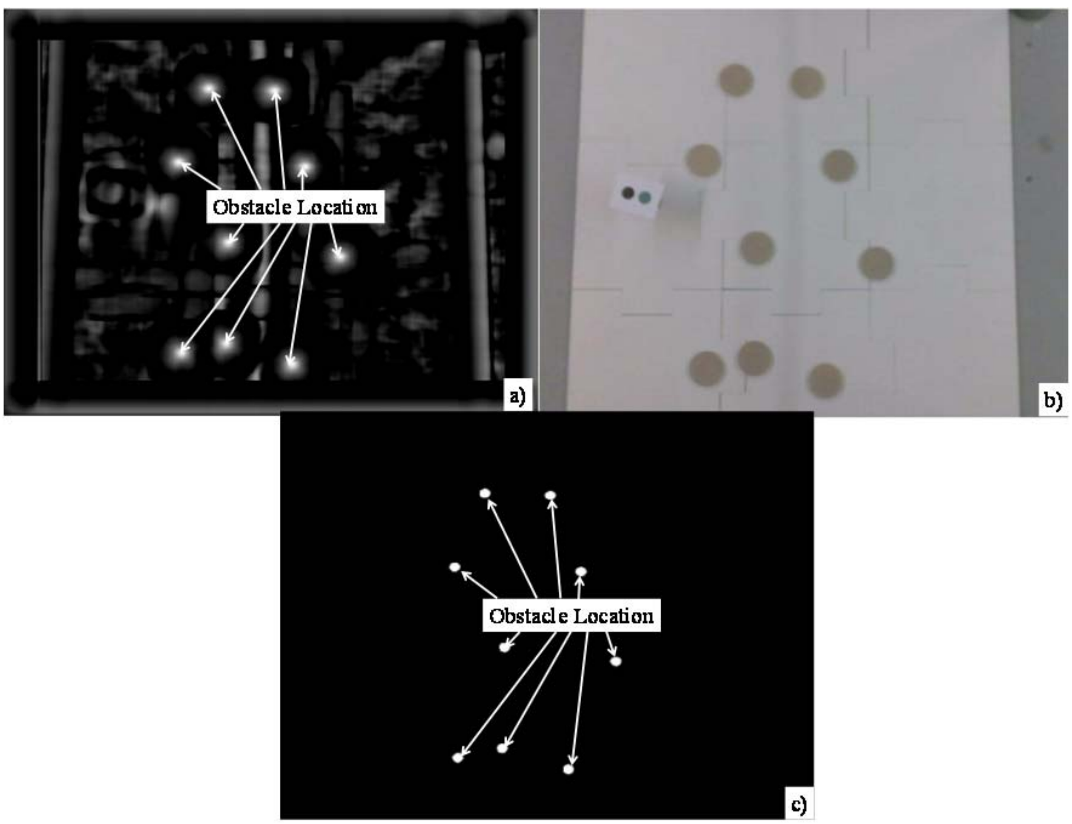

Figure 6. Visual output of a NCC using the obstacle template. a) After pass of NCC b) Original image c) After thresholding

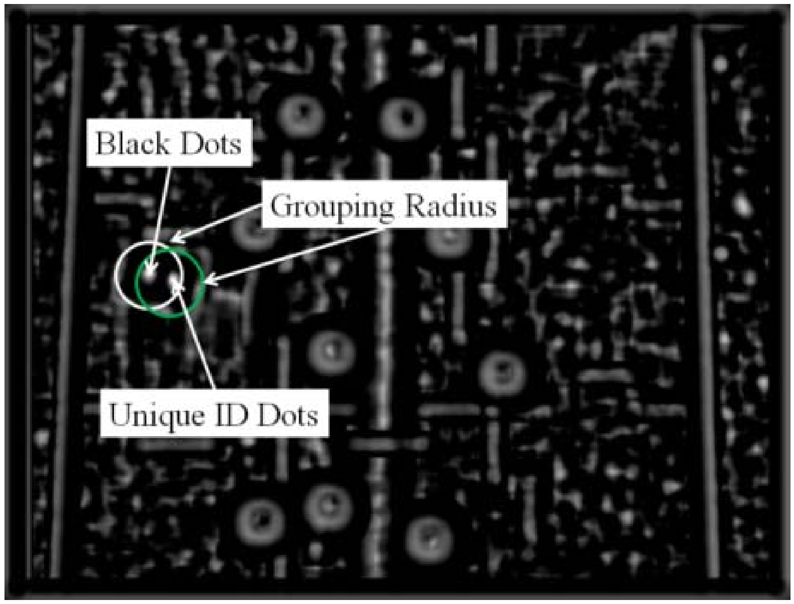

Figure 7. Visual output of a NCC using the robot dot template before thresholding. Based off the same original image seen in Figure 6a.

In order to fully determine the n_r robots' locations and headings, $[x_{rob}, y_{rob}, \theta_{rob}]_i, i = 1,...,n_r$, more computation is required. The average RGB values of the areas surrounding the dot locations, $[x_{dot}, y_{dot}]_i, i = 1,...,2n_r$, are found and compared to the predefined thresholds to determine the sets of unpaired colored dots $x_{cd}, i = 1,...,n_r$, and $x_{bd}, i = 1,...,n_r$ as seen in Figure 7. Black dots and colored dots, the latter of which acts as a unique ID for each robot, are then paired by proximity.

The position for each robot is then determined as

$$\begin{matrix} x_{rob,i} = \left(x_{cd,i} + x_{bd.i}\right)/2 \\ y_{rob,i} = \left(y_{cd,i} + y_{bd.i}\right)/2 \end{matrix}, i = 1,...,n_r \quad , \tag{11}$$

while the heading is found using

$$\theta_{rob,i} = \arctan\left(\frac{y_{cd,i} - y_{bd,i}}{x_{cd,i} - x_{bd,i}}\right), i = 1,...,n_r . \tag{12}$$

When programmed in MATLAB, these algorithms can be used to process images at 100Hz for the robot localization, and 7.4Hz for the obstacle detection, respectively. Therefore all vision data is updated every 0.14 seconds (7.4Hz). The more detailed information about the localization using the developed testbed can be found in [17].

3.7 Path Tracking

Here a tracking controller is designed to calculate proper motor commands for robots to track the generated optimal path $\left(x_i^*, y_i^*\right), i=1,\ldots,n_r$ via the VMC method. If the current position is $(x_{rob,i}, y_{rob,i}), i=1,\ldots,n_r$, the rotation command is given based on

$$\Delta\theta_{rob,i} = \arctan\left(\frac{y_i^* - y_{rob,i}}{x_i^* - x_{rob,i}}\right), i=1,\ldots,n_r, \tag{13}$$

and then a translational command is computed as

$$\Delta x_{rob,i} = \sqrt{\left(x_i^* - x_{rob,i}\right)^2 + \left(y_i^* - y_{rob,i}\right)^2}, i=1,\ldots,n_r. \tag{14}$$

Rotational and translational power commands for a time step Δt can be found by

$$\vartheta_{R,i} = \frac{\Delta\theta_{rob,i}}{\Delta t c_{R,i}}, i=1,\ldots,n_r \tag{15}$$

and

$$\vartheta_{T,i} = \frac{\Delta x_{rob,i}}{\Delta t c_{T,i}}, i=1,\ldots,n_r \tag{16}$$

where

$$c_{R,i} = \frac{r}{d}\left(c_{l,i} + c_{r,i}\right), i=1,\ldots,n_r \tag{17}$$

$$c_{T,i} = \frac{r}{2}\left(c_{l,i} + c_{r,i}\right), i=1,\ldots,n_r \tag{18}$$

This has proven to provide adequate tracking of the planned trajectories, as can be seen in Figure 8, and typically have a maximum tracking error of around 4 pixels or 1.1 cm.

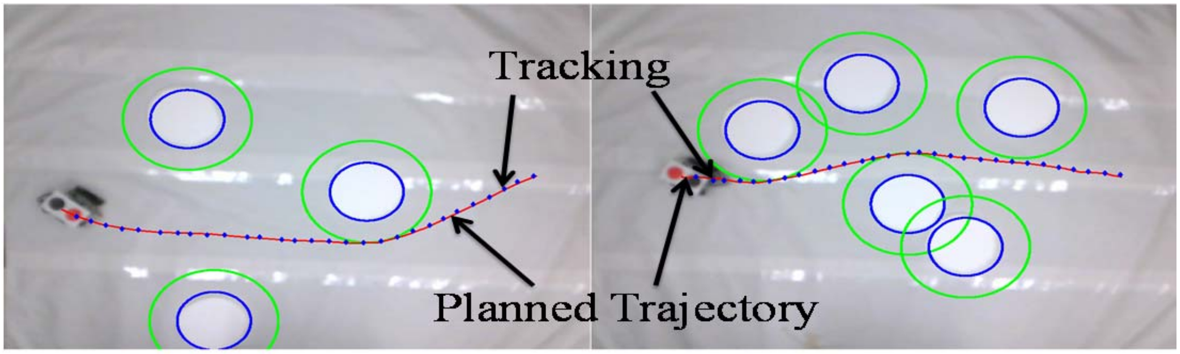

Figure 8. Examples of path tracking (blue diamonds) for given trajectories (red)

3.8 Automatic Robot Parameter Calibration

For a differential drive robot, such as the ones used in the testbed, and assuming a case of no slipping, the robot can be controlled by the turn rate of ω and the translational velocity of V, and the motion is governed by the following dynamics [15]

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos(\theta) \\ \sin(\theta) \\ 0 \end{bmatrix} V + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \omega \tag{19}$$

V and ω, depicted in Figure 9, can be generated based on the left and right wheel angular speeds [16]

$$\begin{bmatrix} V \\ \omega \end{bmatrix} = \begin{bmatrix} \frac{r}{2}(\omega_l + \omega_r) \\ \frac{r}{d}(-\omega_l + \omega_r) \end{bmatrix} \tag{20}$$

with the wheel radius r and the distance between wheels d, measured in cm. The power level commands $[\vartheta_l, \vartheta_r]$, used in the robot firmware, for generating desired angular velocities of the wheels can be computed by

$$\vartheta_i = \omega_i / c_i, \quad i = l, r \tag{21}$$

where the constants c_i, can be found through calibration.

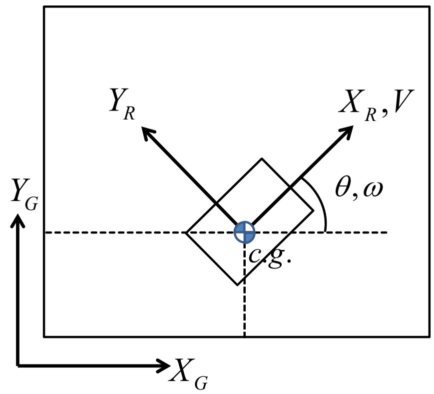

Figure 9. Body and global reference frames used in testbed.

A program was developed for a quick calibration and autonomous calibration. A series of rotation and translation commands are sent to a robot for known time steps and the responses (i.e. $\Delta x, \Delta\theta$) are measured. An example of this can be seen in Figure 10. From these measurements, constants can be found through a least square approach. Table 1 shows the calculated values from the example. It is worth noting that from equations 15 and 16, it isn't necessary to measure the robot's wheel radius and width as they can be grouped into the constants of rotation and translation.

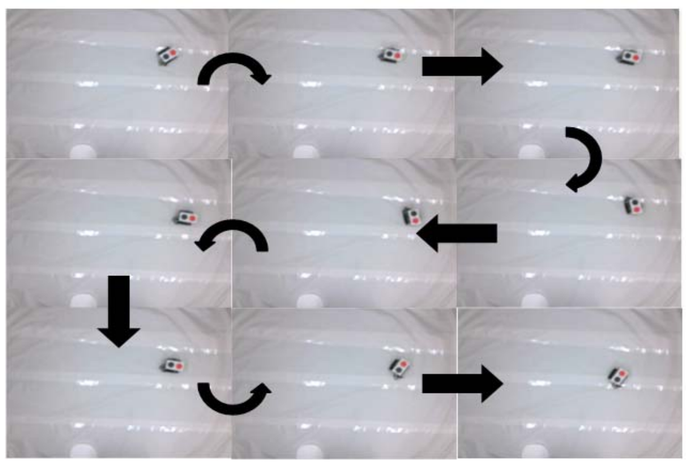

Figure 10: Example of automatic calibration for a robot. Commands alternate between positive and negative rotations and translations.

Table 1: Calculated Values for the Example Calibration

Parameter	Value
r	2.54cm
d	15.24cm
c_r	0.09396
c_l	0.10396

3.9 Software Architecture and Software Package

The software modules developed and the programming languages used are shown in the following table. The main software controls the information flow among sensors, actuators, and all the control algorithms. The information collected from the sensors will be saved in a common bus so that all the software functions can get that information. The main software controls the communication among vision system, Bluetooth on the robots, and the MATLAB files in the laptop. It is worth noting that the names of the software main subroutines for those functionalities are listed in the table and there are many other subroutines have been sent to AFRL-Space Vehicle Directorate.

Table 2: Software Package

Programming Language	Software Module	Main Subroutines' Names
Bottom level algorithms	MATLAB	vmc_ob_top.m
Top level algorithms	MATLAB	Formation_top_prey_alg.m
Communication between the robots and laptops	RORBOTC /MATLAB	Open_Bluetooth.m
Position calculation	MATLAB	CAM_Robust_FindRobots.m
Velocity calculation	MATLAB	CAM_Robust_FindRobots.m
Steering angle calculation	MATLAB	CAM_Heading.m
Steering rate calculation	MATLAB	CAM_Heading.m
Encoder command calculation	MATLAB	Robot_MotorCommand.m
Communication between the laptop and vision system	MATLAB	CAM_Robust_FindRobots.m
Main Software	MATLAB	main_hierarchical.m

A user friendly graphic interface is developed and one screenshot is shown in Figure 11.

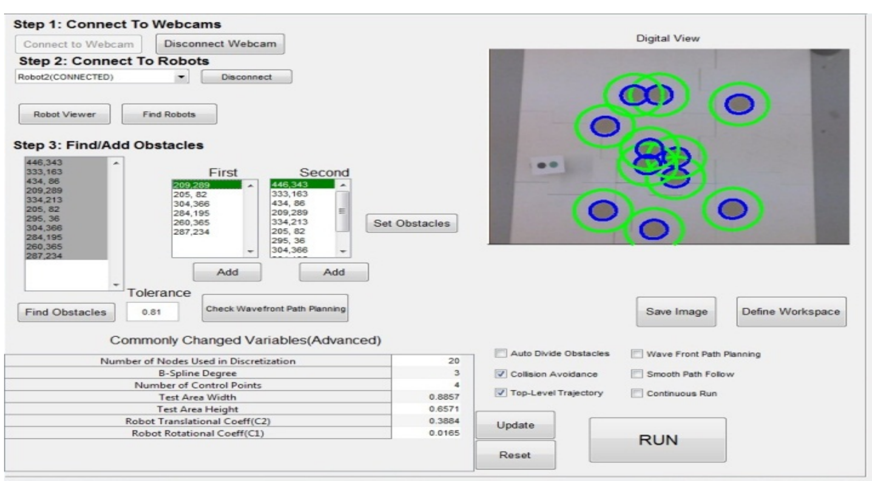

Figure 11. Graphic user interface.

4 RESULTS AND DISCUSSION

4.1 Simulation Results

To test the robustness of the VMC based algorithm, a Monte Carlo simulation is conducted before implemented in the hardware testbed. More than 500 Monte Carlo runs are simulated, in which the initial position, initial velocities, locations of the obstacles, and the sizes of the obstacles are varying. **Figures 12 and 13** are two screenshots from the Monte Carlo simulation. For each case, it takes the MATLAB code roughly 1-5 seconds to obtain the optimal solution for the single vehicle trajectory planning.

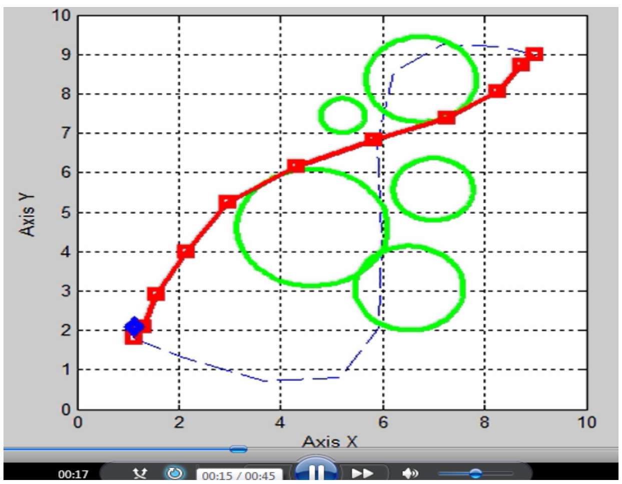

Figure 12. Monte Carlo simulation case 1

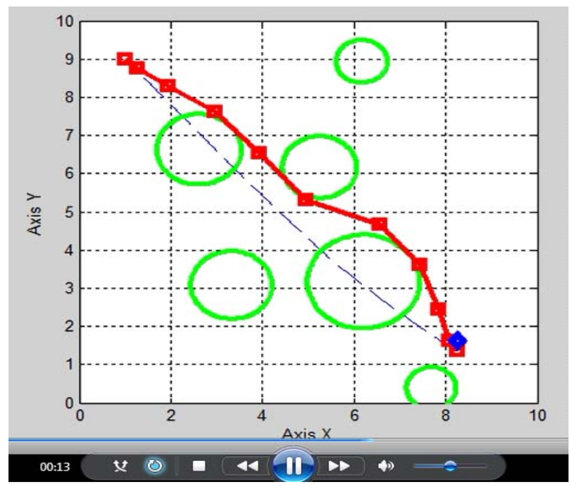

Figure 13. Monte Carlo simulation case 2

4.2 Testbed Demonstration 1 – Single Vehicle Trajectory Planning

The test area and scenario are shown in Fig. 14 through Fig. 17, in which the web camera, laptop, and robots are shown. A couple of collision avoidance minimum time cases are shown here: (1) single robot trajectory planning (Fig. 14), (2) two robots trajectory planning (Fig. 15), (15) the webcam system on the ceiling, and (16) the test area (100 inches by 72 inches)

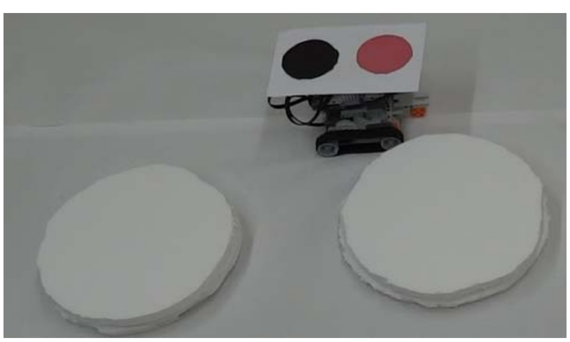

Figure 14. Single robot planning

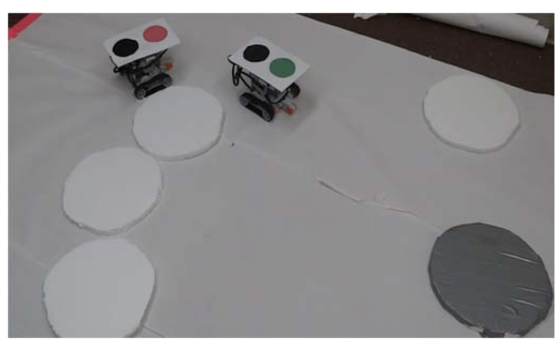

Figure 15. Two robots planning

Figure 16. Webcam on the ceiling

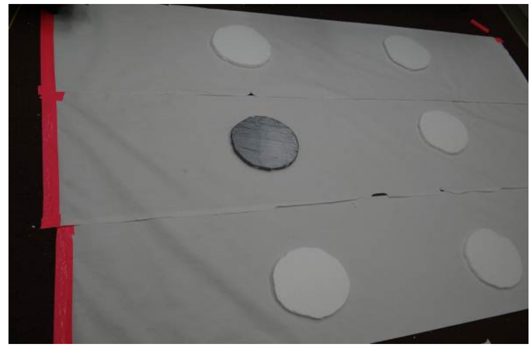

Figure 17. Testbed area

Figure 18 shows one trajectory planning and re-planning results using the VMC method in the hardware test. As it is shown, when two new obstacles are popped up in the path, the VMC algorithm quickly generated a different optimal path considering these two obstacles.

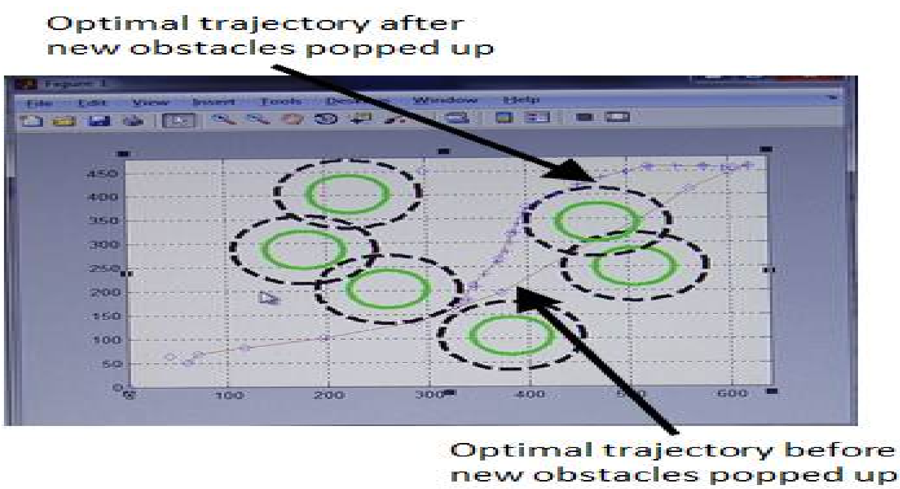

Figure 18. Trajectory planning (before certain obstacles shown up) and re-planning (after certain obstacles shown up)

4.3 Testbed Demonstration 2 – Hierarchical Cooperative Control

The results of three separate runs in testbed at UCF, using the proposed hierarchical cooperative control method, are shown. Each formation was driven to the target pixel location of [550, 240], but each robot's position relative to the formation's center was changed for each run.

Case 1 (Fig. 19): Robots 1 and 2 started at pixel locations of [50, 186] and [58, 293], respectively. Given the desired formation positions of [0, -80] and [0, 80], the initial robot trajectories took 0.22 seconds to compute. The optimization of the trajectories took 7.35 seconds to converge to for robot 1 and 11.03 seconds for robot 2.

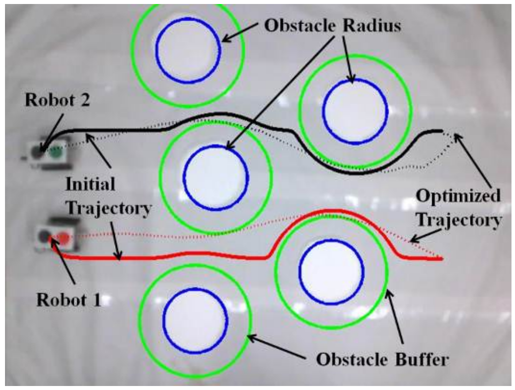

Figure 19. Hierarchical cooperative trajectory planning case 1

Case 2 (Fig. 20): Robots 1 and 2 started at pixel locations [89, 424] and [92, 50], respectively. Given the desired formation positions of [0, 50] and [0, -50], the initial robot trajectories took 0.23 seconds to compute. The optimization of the trajectories took 8.91 seconds to converge to for robot 1 and 18.34 seconds for robot 2.

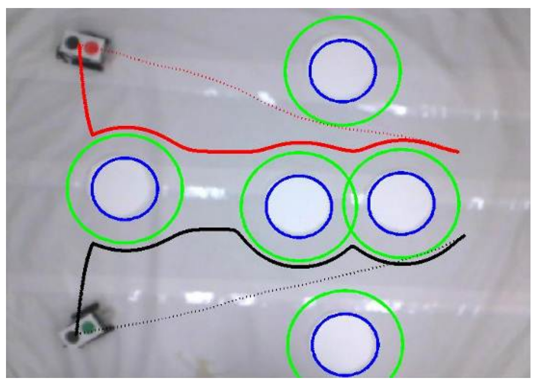
Figure 20. Hierarchical cooperative trajectory planning case 2

Case 3 (Fig. 21): Robots 1 and 2 started at pixel locations [94, 95] and [129, 394], respectively. Given the desired formation positions of [0, 50] and [0, -50], the initial robot trajectories took 0.15 seconds to compute. The optimization of the trajectories took 6.01 seconds to converge to for robot 1 and 18.93 seconds for robot 2.

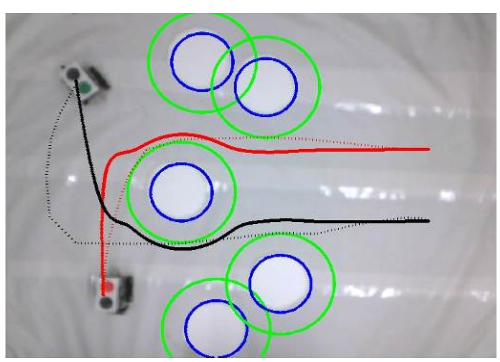
Figure 21. Hierarchical cooperative trajectory planning case 3

5 CONCLUSIONS

A low cost, vision based robot testbed is designed and implemented for the purpose of validating new trajectory planning algorithms for single- or multi-vehicle systems. This testbed can support multiple structurally changeable robot platforms and uses an overhead vision system for robot localization and obstacle detection. An easy-to-use graphical user interface is designed to assist users in simulating a complex environment and quickly changing the algorithms being tested. A specific algorithm, the B-Spline augmented virtual motion camouflage method enhanced by the wavefront path planning algorithm, is demonstrated in this testbed for robots to navigate through an obstacle dense environment. Results show that the testbed developed is functioning properly and the optimal trajectories can be found rapidly while avoiding obstacle and inter-robot collisions.

The research results have generated the following publications.

- G. Basset, Y. Xu, and K. Pham, "Bio-Inspired Rendezvous Strategies and Respondent Detections," Accepted to the *AIAA Journal of Guidance, Control, and Dynamics*, May 2012.

- Basset, G., Xu, Y., and Pham, K. D., "Motion Camouflage Feasibility and Detection for Space Situational Awareness," *2012 American Control Conference*.

- G. Basset, R. SiVilli, Y. Xu, and K. Pham, "Minimum-Time Obstacle Avoidance Trajectory Planning for Vision Based Robots," submitted to *Robotica*.

- R. SiVilli, Y. Xu, and K. Pham, "A Vision-Based Robot Testbed for Single or Multiple Vehicles' Trajectory Planning," submitted to *2013 American Control Conference*.

The Software Version 2.0 (Hierarchical Cooperative Control) has been delivered to AFRL-RV. Also the hardware testbed video has been delivered to AFRL-RV.

Some potential research and development improvements include: (i) A cooperative trajectory planning algorithm, based on the modified local pursuit strategy, needs to be investigated for networked spacecraft in proximity operations, which can handle different constraints, nonlinear dynamics, conflict resolution, and obstacle avoidance. (ii) A system identification method needs to be used online to quantify the coefficients of the nonlinear robot dynamic model. (iii) An upgraded testbed should be designed, integrated, and tested at AFRL-RV. (iv) A quad-rotor can be used to hover around the testbed, on board of which a web camera and an altimeter will provide real-time information about the test area and moving targets in the proximity operations (e.g. rendezvous maneuvers). A vision algorithm considering the moving platform should be investigated. The testbed needs to be more flexible so that other applications (e.g. game problems) can be tested.

REFERENCES

[1] NSF CPS Program Synopsis, http://www.nsf.gov/pubs/2010/nsf10515/nsf10515.htm, last access on March 10, 2010.

[2] Ren, W., and Beard, R. W., *Distributed Consensus in Multi-Vehicle Cooperative Control – Theory and Application*, Springer-Verlag, London, 2008.

[3] Dong, W. J., and Farrell, J. A., "Cooperative Control of Multiple Nonholonomic Mobile Agents," *IEEE Transactions on Automatic Control*, Vol. 53, No. 6, July 2008, pp. 1434-1448.

[4] Ren, W., "Second-order Consensus Algorithm with Extensions to Switching Topologies and Reference Models," *American Control Conference*, July 9-13, 2007, New York, NY, pp. 1431-1436.

[5] Dunbar, W. B., and Murray, R. M., "Model Predictive Control of Coordinated Multi-vehicle Formation," *41th IEEE Conference on Decision and Control*, Las Vegas, NV, December 10-13, 2002, pp. 4631-4636.

[6] Fahroo, F and Ross, I. M., "Costate Estimation by a Legendre Pseudospectral Method," *Journal of Guidance, Control, and Dynamics*, Vol. 24, No. 2, March-April 2001, pp. 270-275.

[7] Huntington, G. T., and Rao, A. V., "Optimal Reconfiguration of Spacecraft Formations using the Gauss Pseudospectral Method," *Journal of Guidance, Control, and Dynamics*, Vol. 31, No. 3, May-June 2008, pp. 689-698.

[8] Yunjun Xu, Ming Xin, Jianan Wang, and Suhada Jayasuriya, "Hierarchical Control of Cooperative Nonlinear Dynamical Systems," *International Journal of Control*, Vol. 85, No. 8, 2012, pp. 1093-1111.

[9] G. Basset, Y. Xu, and N. Li, "Fast Trajectory Planning via the B-Spline Augmented Virtual Motion Camouflage Approach," *50th IEEE Conference on Decision and Control*, Orlando, FL, Dec. 2011.

[10] Y. Xu, G. Basset, "Virtual Motion Camouflage based Phantom Track Generation through Cooperative Electronic Combat Air Vehicles," *Automatica*, Vol. 46, Issue 9, September 2010, pp. 1454-1461.

[11] Srinivasan, M. V., and Davey, M., "Strategies for active camouflage motion," *Proceedings of the Royal Society of London Biological Sciences*, Vol. 259, No. 1354, 1995, pp. 19-25.

[12] Xu, Y., "Motion camouflage and constrained suboptimal trajectory control," *2007 AIAA Guidance, Control, and Dynamics Conference*, August 20-23, 2007, Hilton Head, South Carolina.

[13] Piegl, L., and Tiller, W., *The NURBS Book: Second Edition*, Springer-Verlag, New York, 1997.

[14] Laumond, J. P., Sekhavat, S., and Lamiraux, F., "Guidelines in nonholonomic motion planning for mobile robots," *Lecture Notes in Control and Information Sciences*, LNCIS 229, New York: Springer-Verlag, 1998, pp. 1-44.

[15] J. P. Laumond, S. Sekhavat, and F. Lamiraux, *Robot motion planning and control*. LASS-CNRS, Toulouse: LAAS, 1998.

[16] K. Park, H. Chung, J. Choi, and J. G. Lee, "Dead reckoning navigation for an autonomous mobile robot using a differential encoder and a gyroscope," presented at the ICAR '97, Monterey, CA, 1997.

[17] R. SiVilli, Y. Xu, and K. Pham, “A Vision-Based Robot Testbed for Single or Multiple Vehicles’ Trajectory Planning,” submitted to 2013 American Control Conference.
[18] K. Briechle and U. Hanebeck, “Template Matching Using Fast Normalized Cross Correlation,” Proceedings of SPIE, Vol. 4387, April 2001.

LIST OF ACRONYMS, ABBREVIATIONS, AND SYMBOLS

ACRONYM	Description
MC	Motion Camouflage
VMC	Virtual Motion Camouflage
PCP	Path Control Parameter
NCC	Normalized Cross Correlation
RGB	Red, Green, and Blue
UCF	University of Central Florida

DISTRIBUTION LIST

DTIC/OCP 8725 John J. Kingman Rd, Suite 0944 Ft Belvoir, VA 22060-6218	1 cy
AFRL/RVIL Kirtland AFB, NM 87117-5776	2 cys
Official Record Copy AFRL/RVSV/Khanh Pham	1 cy